Using buddies to live longer in a boring world

Samir Goel[1], Andrea Passarella[2], and Tomasz Imielinski[1]
gsamir@cs.rutgers.edu, a.passarella@iet.unipi.it, imielins@cs.rutgers.edu

[1]Department of Computer Science
Rutgers University, USA

[2]Department of Information Engineering
University of Pisa, Italy

Rutgers Department of Computer Science Technical Report
DCS-TR-558

September, 2004

Abstract

In a sensor network monitoring natural environment, the readings of sensor nodes show high temporal and spatial correlation. This stems from the fact that most characteristics of natural environment (chemical concentration, temperature, humidity, pollution, etc) do not change abruptly in space and time. From a sensor's perspective then, the environmental phenomena are quite predictable, and hence, "boring". In this paper, we introduce the novel idea of *buddies* to exploit temporal correlation. We propose a simple Buddy protocol to implement this idea. We analyze the performance of the protocol and identify factors that influence it. We discuss research challenges that would pave the way for more efficient protocols based on this idea.

1 Introduction

Numerous applications of sensor networks require sensor nodes, operating on AA batteries, to last for months if not years. Examples include habitat monitoring, battlefield reconnaissance, forest-fire detection and notification, chemical concentration sensing, etc. Today's batteries cannot support lifetime of months/years — a Berkeley Mote with all its components (sensors/radio) ON would last for 30 hrs on a single Energizer CR2450 lithium battery [5]. Going by the current technology trend, the battery capacity is expected to improve at a slow rate of 2-3% per year [6]. Thus, the only way to extend lifetime of a sensor node is by using intelligent mechanisms that make judicious use of the energy resources.

It is a well-known fact that radio is the most energy consuming resource in a sensor node [5]. Energy cost of transmitting/receiving one byte is at least an order of magnitude higher than that for sensing and computation[1]. At a higher level, it has been shown [10] that *idle listening*, the case when radio is ON and idle, waiting for communication from its neighbors, is the dominant factor in energy consumption. This suggests that a sensor node should not only transmit as few bytes as possible, it should also turn OFF its radio as much as possible.

In this paper, we propose to achieve this objective by exploiting the temporal correlation in the readings of sensor nodes. Our key idea is that two neighboring nodes can help reduce each other's energy consumption by entering into a collaborative *buddy* relationship. These *buddies* take turns in keeping their radio ON. At any point of time, the node that has its radio ON also act as a representative for its buddy, answering queries and participating in monitoring operation on its behalf. Buddies exchange information in order to

[1]The cost of transmitting one byte is approximately the same as that for receiving it [6, 2].

make sure that the representative node's response on behalf of its buddy meets the accuracy constraint. In the ideal case very little information needs to be exchanged. In such a case, this collaborative arrangement would allow both nodes to cut down their energy consumption by a factor of 2.

This basic idea can be extended to a group of N collaborating nodes. These nodes take turns to keep their radio ON. The representative node now responds on behalf of its $N-1$ buddies. In the ideal case, this cuts down the energy consumption by a factor of N. At a conceptual level, when all sensor nodes act individually, they can be seen as burning their batteries in parallel. Buddy protocol may be viewed as an attempt to serialize the burning of batteries of these sensor nodes, as much as possible, thereby extending their lifetime.

While saving energy, Buddy protocol also ensures that *three* key characteristics of the network are maintained: (a) any degradation in the quality of monitoring operation is bounded by network or application-specified constraints, (b) nodes meets the constraints on per-hop communication delay, and (c) turning OFF of node's radio does not adversely impact the reachability of the network.

These three characteristics differentiate this work from existing work on distributed clustering protocols (e.g., GAF [9], CEC [8], etc). Such protocols divide the nodes in the network into clusters, such that even if any one node from each cluster is ON, the reachability in the network is preserved. They save energy by turning OFF radios of nodes that are redundant with respect to connectivity. However such solutions do not consider the fact that while nodes may be redundant with respect to connectivity, their readings may not be redundant. These solutions save energy at the cost of *unbounded* reduction in quality of monitoring. Buddy protocol attempts to save energy while at the same time meeting the constraint on quality of monitoring.

Temporal correlation may also be exploited by using local compression at the sensor node. Buddy protocol differs from such schemes in an important way. With compression a node necessarily needs to delay transmission of its readings. The larger the delay, the higher the potential for reduction in the size of transmission. Such delay may not be acceptable to the application. For example, in the setup of Great Duck Island Project [7], sensor nodes transmitted their readings once every 5 minutes. In order to compress readings from, say, 10 epochs, a sensor node would need to delay its response by 50 minutes!

In this paper, we describe the architecture and the protocol that implements the idea presented above. Buddy protocol is based on *prediction-based monitoring* (PREMON) [3] paradigm. We analyze the performance of the protocol and identify factors that influence it.

In the next section, we describe our assumptions about the system. In section 3, we briefly review PREMON paradigm. In section 4, we provide details of the Buddy protocol. In the subsequent section, we analyze the performance of Buddy protocol and identify factors that influence it. We enumerate the research challenges in section 6, and conclude in section 7.

2 System Model

We consider a multi-hop sensor network that supports monitoring operation. A monitoring operation requires sensor nodes to report their readings every T_a units of time (e.g., "Report pollution level in the tunnel, every 5 minutes"). We assume that with each query or monitoring operation, the application specifies an error threshold, γ. This defines the maximum acceptable difference between the readings received by the application and the true readings of the sensor. We also assume that all nodes must guarantee a maximum per-hop communication delay of T_d.

3 Background

3.1 PREdiction-based MONitoring paradigm

PREMON [3] exploits correlation in sensor readings in order to save energy. Consider a sensor node being monitored. Instead of reporting readings periodically, a sensor node in PREMON generates a prediction of its readings, encodes it concisely as a *prediction-model*, and sends it to the monitoring entity. The sensor

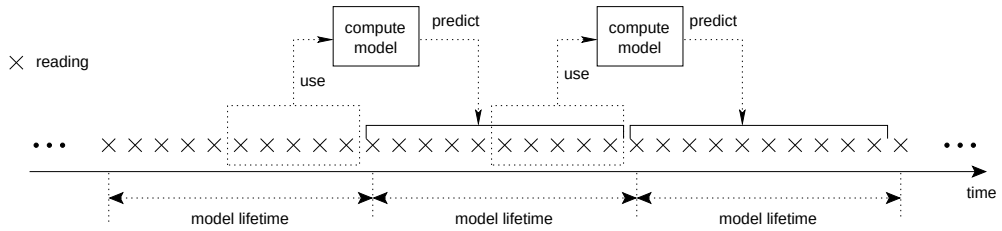


Figure 1: Mechanism for computing prediction

node also specifies the duration for which the prediction model is valid. During this lifetime, the sensor node transmits only those readings that differ from the predicted readings by more than a certain pre-specified *error threshold*, γ. Such readings are termed as *"violations"*. In the absence of any transmissions from the sensor node, the monitoring entity uses the sensor's prediction model to determine the readings of the sensor node. At the end of the lifetime of the prediction-model, the sensor generates a new prediction model and sends it to the monitoring entity.

Note that in PREMON a monitored sensor node should keep its radio ON even during intervals when its readings are predictable. This allows it to respond to requests from other monitoring entities in the network.

PREMON paradigm trades increased computation (for deriving prediction models) for savings in number of transmissions. For Berkeley Mote, the energy cost of transmitting one packet of 30 bytes is 240 μJ (transmission cost is $1\mu J$/bit [5]). With this energy, one can perform 30,000 machine instructions (cost of performing 100 machine instructions is approximately 0.8 μJ [5]). Given these relative costs, in general, we expect that saving of even a few transmissions would more than compensate for the cost of computing prediction model.

Computing a good prediction model an orthogonal problem. Based on the characteristics of the application, the network designer may pre-specify a set of basis functions, $F = \{f_1(t), f_2(t), ..., f_n(t)\}$. The prediction model, $M(t)$, may then be concisely expressed as a vector of coefficients, $W = \{w_1, w_2, ..., w_n\}$, such that:

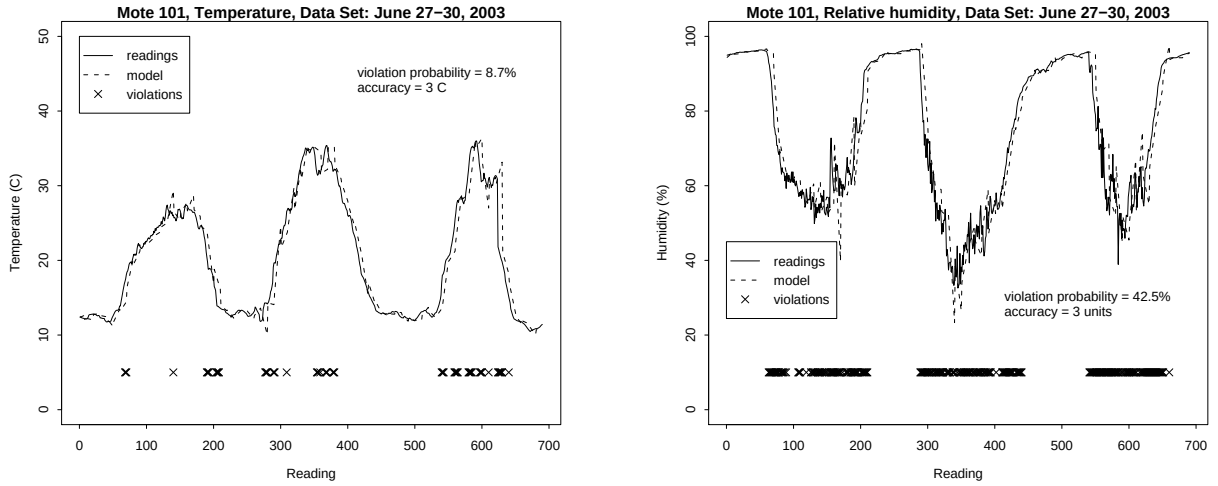
$$M(t) = w_1 f_1(t) + w_2 f_2(t) + ... + w_n f_n(t) \quad (1)$$

where, $t = t_1$, $t_1 + T_a$, $t_1 + 2T_a$, ..., t_2, and T_a is the reporting frequency requested by the application. $M(t)$ is defined only at these discrete instants of time; at all other points, its value is undefined. At points where $M(t)$ is defined, the sensor node makes sure that the prediction model is accurate to within γ (by sending violation, if required). Computing prediction-model may involve simple linear regression, or it may use sophisticated regression techniques [4].

3.2 How predictable is data in real world?

We analyzed the dataset collected during the Great Duck Island Project [7]. We present the results for humidity and temperature readings from one sensor node. The samples were collected every 5 minutes for approximately 3 days.

We use linear regression for generating predictions. Specifically, we use K readings of the sensor to estimate the coefficients of the linear model, $M(t) = w_1 + w_2 \cdot t$ that results in best fit (minimum mean square error). We use the resulting model as a predictor for the next Δ readings. After Δ readings, the value of coefficients $w_1$and w_2 of the model are estimated once again, using the last K readings. Figure 1 captures this mechanism. Figure 2 shows the result of applying this technique to the readings of humidity and temperature sensor. The parameter values chosen were: $K = 5$, $\Delta = 10$, and error threshold $\gamma = 3$ units (for both sensors). The results show that this simple prediction model "correctly" predicted 91% of the readings of temperature sensor, and 57% of the readings for humidity sensor. This example illustrates how a simple prediction model can save more than 50% transmissions from the sensor node for a small value



(a) Temperature sensor

(b) Humidity sensor

Figure 2: Performance of simple predictor over real world data

of γ. Figure 3 shows the performance as a function of error threshold, γ. The percentage of "incorrect" predictions (violations) drop substantially with the increase in γ. Thus, PREMON allows an application to trade accuracy of monitoring operation for reduction in the number of transmissions.

4 Buddy Protocol

4.1 Basic Idea

In order to exploit correlation in its readings, each node attempts to establish buddy relationship with its neighbors. This results in formation of a number of buddy-groups in the network. Each buddy-group has one representative node responsible for answering queries and participating in monitoring operation on behalf of all the other nodes in the group. In the ideal case, this allows all other nodes to turn their radio OFF, resulting in significant energy savings. The members of the buddy-group collaborate to rotate the responsibility of being the representative, so as to spread the consumption of energy uniformly, over the group members.

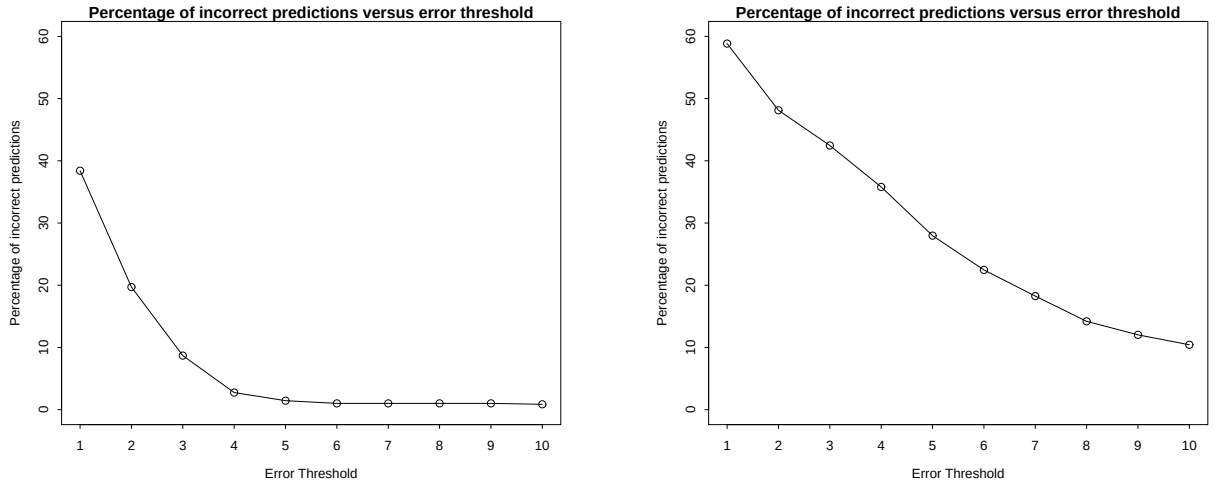
Two key characteristics must be preserved in order to make this energy-saving mechanism transparent to the application:

- Quality of monitoring
- Reachability in the network

We describe how these two characteristics are preserved.

4.2 Preserving quality of monitoring

The formation of buddy-groups should not result in unacceptable level of degradation in the quality of results returned by the network (in response to queries or monitoring operation). In order for the representative node to respond on behalf of other nodes in the buddy-group and still meet this constraint on quality, it *must*



(a) Temperature sensor

(b) Humidity sensor

Figure 3: Performance of simple predictor as a function of error threshold, γ

know their readings accurate to within γ. A straight forward approach to meet this requirement would be to have all nodes in the buddy-group treat the representative node as a cache (or proxy) and keep its state "consistent" by sending updates. A more general and promising approach is to use PREMON paradigm. With this approach, the nodes may send their prediction model to the representative node. Subsequently, they only report violations to the representative node. When the readings of the node are highly correlated, this would lead to substantial reduction in transmissions. Note that as long as the prediction model at the representative node is close to the actual readings of the sensor (i.e., sensor's readings are predictable), it can switch OFF its radio. This cuts down idle listening, a dominant factor in energy consumption at a sensor node [10].

Clearly, using PREMON mode does not always result in energy savings. During the lifetime of a sensor node, there are intervals where the readings are highly unpredictable (for example, the readings of magnetometer sensor in battlefield as a tank passes by). These unpredictable readings represent "interesting" events, and in many cases, are precisely the ones that are of interest to the network users. During such intervals, it may be more energy efficient to operate in DEFAULT mode, wherein the node reports readings at regular intervals. Buddy protocol allows a node to intelligently choose its mode of operation so as to save energy.

4.3 Preserving reachability in the network

Energy savings increase when fewer nodes keep their radio ON. This argues for fewer buddy-groups of higher cardinality. However, when large number of nodes turn their radio OFF, the reachability in the network may be affected. For example, in a chain network, if any of the intermediate nodes turns its radio OFF, it would partition the network. The goal of preserving reachability influences the way buddy-groups may be formed. Complicating matters further is the fact that a node's impact on reachability in the network is dependent on the decision of its neighbors. There is a need for a simple distributed mechanism that would guide the formation of buddy relationships while preserving the "reachability" in the network.

One possible solution is to make use of one of the distributed clustering algorithms (e.g., GAF [9], CEC [8], etc). The goal of such an algorithm is to divide the nodes in the network into clusters so that even when any

one node from each cluster has its radio ON, the reachability of the network is still preserved. In this paper, we assume that GAF [9] is used for clustering. It is important to note that GAF is simply a placeholder, and can be replaced by any other clustering scheme that is more suitable for the environment at hand.

With GAF all nodes within a cluster are equivalent with respect to connectivity, and only one of them need to keep their radio ON. The node that takes up this responsibility is called the *cluster-head*, and is elected by the nodes using a cluster-head election algorithm. Cluster-head election algorithm is run periodically in order to rotate this responsibility among nodes in the cluster, thereby spreading the energy consumption uniformly over all the nodes in the cluster.

From the point of view of the Buddy protocol, each cluster is a buddy-group, and the cluster-head is the representative node for the group.

4.4 Buddy Protocol: Putting it all together

A distributed clustering protocol is used to group the nodes in clusters. The set of cluster-heads form the routing backbone of the network. Within a cluster, we assume that nodes follow a style of operation similar to PSM (power-saving mode) mode of 802.11 [1]. The cluster-head performs the role of access-point, buffering data for all the nodes in the cluster. The cluster-head indicates the presence of buffered data by using MAC-level beacons that are sent out periodically. In order for nodes to meet the per-hop communication delay requirements, the nodes wake up every T_d units of time to receive beacon from the cluster-head. We assume that T_d is an integer multiple of the beacon-period.

Within each cluster, every node (except cluster-head) estimates the cost of operating in PREMON mode and DEFAULT mode. Every T_Δ units of time ($T_\Delta = \Delta \cdot T_a$), the node decides the mode of operation that is going to be more energy efficient, and switches to that mode. It stays in the chosen mode for T_Δ units of time. If a node decides to use PREMON mode, it sends prediction-model of its readings to the cluster-head. The node may then turn OFF its radio, turning it ON only for sending violations. We assume that all messages between cluster nodes and the cluster-head are sent reliably. If a node decides to operate in DEFAULT mode, it need not send any message to the cluster head. The cluster-head knows the mode of operation of all the nodes in the cluster. It responds on behalf of the nodes operating in PREMON mode. It passes queries/monitoring operation request to any node operating in DEFAULT mode. These nodes report their readings directly to the querying/monitoring entity.

In the next section, we describe how a node may estimate the cost of operating in DEFAULT and PREMON mode.

5 Performance Analysis

In this section we derive expression for the cost of operation in the two modes. We identify factors that influence these costs and derive "feasible regions" in the parameter space where using PREMON mode is more cost effective than DEFAULT mode. We express the energy cost in terms of the duration for which the radio was ON. This is a good approximation for two reasons: (a) Energy consumed by radio is an order of magnitude higher than that of computation, and (b) The cost of transmitting and receiving data are roughly the same [6, 2].

5.1 Estimating cost in DEFAULT mode

When a node is not being monitored, it only needs to listen to beacons from the cluster-head every T_d units of time. The costs of doing this over the time interval T_Δ is given by:

$$T_D^{NM} = \frac{T_\Delta}{T_d} \cdot T_{beac} \tag{2}$$

where, T_{beac} is the duration of a beacon packet. A node that is being monitored needs to also send its readings every T_a units of time. The cost of doing this over the time interval T_Δ is given by:

$$T_D^M = \frac{T_\Delta}{T_d} \cdot T_{beac} + \frac{T_\Delta}{T_a} \cdot T(d) \tag{3}$$

where, $T(d)$ represents the average time required to send a packet, of size d bits, reliably to the cluster-head. It is a function of channel bit error rate, BER, and packet size.

Note that all parameters required to compute the cost of operating in DEFAULT mode, for T_Δ units of time, are available locally at the sensor node.

5.2 Estimating cost in PREMON mode

As in DEFAULT mode the node listens to the beacon from the the cluster-head every T_d units of time. Every T_Δ units of time, the node generates a prediction-model. This model is sent to the cluster-head. Every T_a units of time, the node compares its readings with the one given by the prediction-model. If these two differ by more than γ, the error threshold, the node declares its current reading as *violation* and sends it to the cluster-head. Let p_v represent the probability of occurrence of this event. Then, during time interval T_Δ, on an average, the sensor node needs to transmit $p_v \cdot \Delta$ number of readings to the cluster-head. The total cost of operating in PREMON mode is given by:

$$\begin{aligned} T_B \quad = \quad & \{T(m) + T_{beac} + p_v \cdot \Delta \cdot (T_{beac} + T(d))\} \\ & + \{(\tfrac{T_\Delta}{T_d} - \Delta) \cdot T_{beac}\} \end{aligned} \tag{4}$$

Note that the first part of the above equation represents the cost of sending a prediction-model (of size m bits) and sending violations. The second part represents the cost of listening to the beacons from the cluster-head. Note that every T_a units of time, the node can skip listening to the beacon because for this epoch the cluster-head knows the reading of the sensor node. Also note that unlike DEFAULT mode, the energy cost in PREMON mode is the same irrespective of whether the sensor node is being monitored or not.

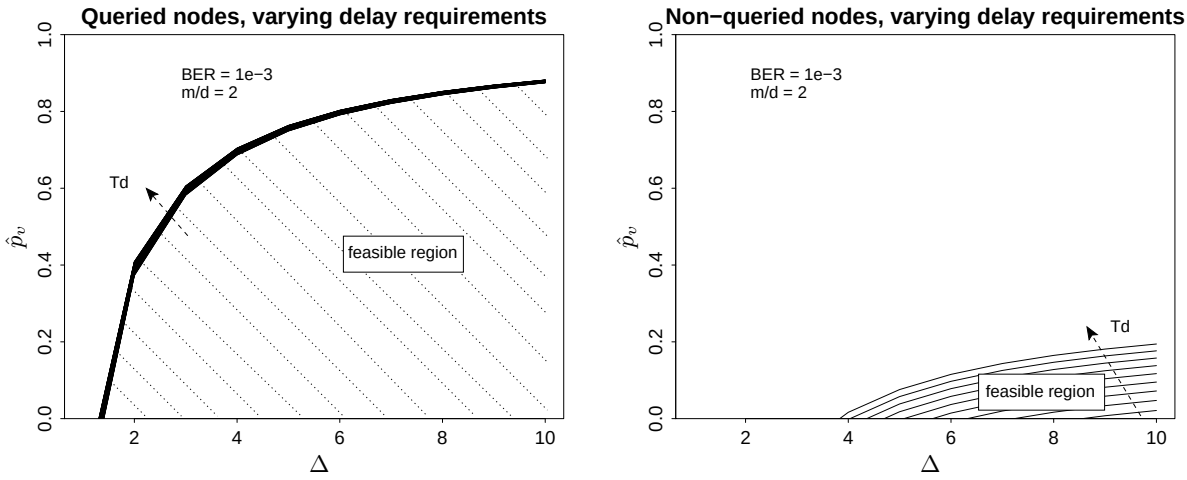
We have ignored the cost of computing prediction model from equation 4 because of the difficulty of estimating it in the general case. Given the relative cost of communicating versus cost of performing one machine instruction (section 3.1), we expect the cost in equation 4 to be a good approximation in most cases.

For computing the energy cost of PREMON mode all parameters except p_v are known locally at the sensor node. Sensor node keeps a running estimate of p_v, by locally generating a prediction-model and comparing its actual readings against the predicted ones. It does this irrespective of its mode of operation.

5.3 Feasible Region

We analyze the energy costs in the two modes in order to identify "feasible regions" in the parameter space where PREMON mode is more energy efficient than the DEFAULT mode. Let I_{ps} be the ratio of energy cost in PREMON mode to that in DEFAULT mode. Let us define $\hat{p}_v$ as the value of p_v for which $I_{ps} = 1$. Thus $\hat{p}_v$ defines range of $p_v = [0, \hat{p}_v)$ for which PREMON mode is more energy efficient (i.e., the boundary of the feasible region).

Figure 4(a) shows result for the case when node is being monitored. It shows that the feasible region in the parameter space for p_v and Δ is quite large. Results show that $\hat{p}_v$ increases with increase in Δ — the increase is sharp for low values of Δ, and it flattens out for larger values of Δ. This is to be expected because with the increase in Δ, the extra overhead of transmitting the prediction-model gets amortized over larger lifetime of the prediction-model. The flattening out of graph for larger values of Δ has important consequences — a prediction model need *not* predict correctly too far into the future to be effective. It argues for simple prediction-models with good prediction accuracy in the short-term. Also, note that as Δ increases, PREMON mode becomes more energy efficient even for nodes with less predictable readings (e.g., for $\Delta > 4$, $p_v = 0.6$ also falls in feasible region). Note that the feasible region is largely insensitive to the value of T_d. This is because of the small cost of listening to the beacons as against that for transmitting readings.



(a) Results fot node being monitored

(b) Results for node not being monitored

Figure 4: $\hat{p}_v$ as a function of Δ

Figure 4(b) shows result for the case when the node is not being monitored. The feasible region is small and is very sensitive to the value of T_d. This is to be expected because the cost in PREMON mode remains the same, whereas the cost in DEFAULT mode is substantially smaller — the only operation involved is listening to beacons from the cluster-head every T_d units of time. Thus, in general, when a node is not being monitored, it is more energy efficient to operate in DEFAULT mode.

5.4 Energy savings in PREMON mode

We now focus on nodes that are being monitored, and analyze the impact of various parameters on the cost of PREMON mode. We eliminate the effect of polling on the cost by assuming that $T_d = T_a$.

5.4.1 Effect of accuracy of prediction model

Figure 5 shows I_{ps} as a function of p_v for different values of Δ. Linear increase in I_{ps} indicates that the energy consumption in PREMON mode increases linearly with p_v. As expected, increase in Δ reduces the energy consumption in PREMON mode. The figure 5 shows that, in the best case, for the parameters chosen, energy cost in PREMON mode is 10% of that in DEFAULT mode — a reduction by a factor of 10!

5.4.2 Effect of size of prediction-model and BER

We capture the effect of relative size by the ratio $\frac{m}{d}$ — the ratio of the size of packet containing prediction-model to that containing the reading. Figure 6 shows the dependence of I_{ps} on $\frac{m}{d}$ and BER on the wireless link. The results were generated for $p_v = 0$ — perfectly predictable sensor readings. As the ratio increases, the energy required to deliver the prediction-model increases. As BER increases the difference in energy cost for delivering the two types of packets becomes more pronounced. Interestingly, the results show that even for perfectly predictable sensor readings, the PREMON mode may cost more energy than DEFAULT mode for certain range of values of BER and $\frac{m}{d}$. As p_v increases, although the shape of curves remain the same, the set of curves move up (we have omitted this graph for lack of space). Thus, it is important to strike a

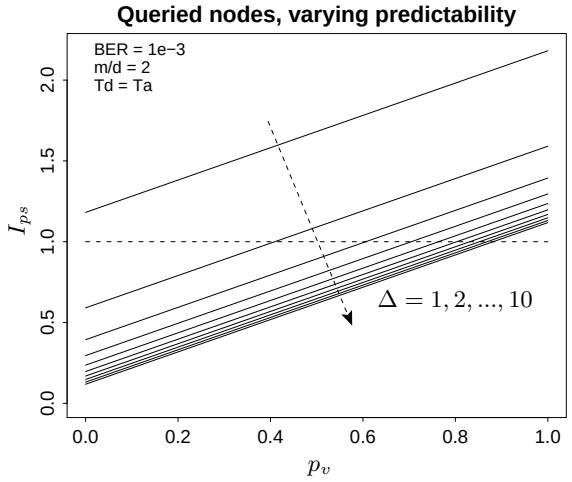


Figure 5: Energy savings as function of accuracy of the model

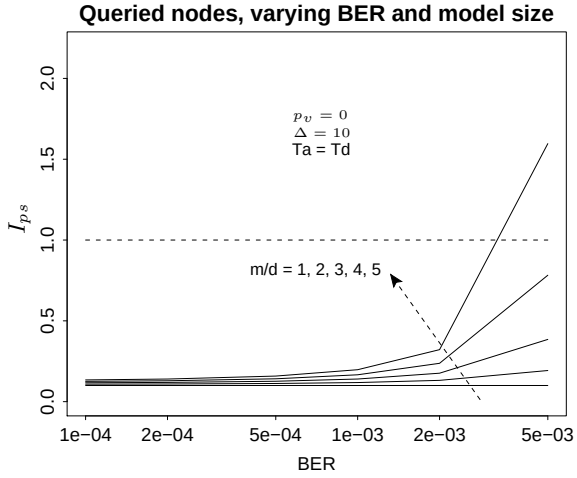


Figure 6: I_{ps} as a function of the BER and $\frac{m}{d}$

good balance between size of prediction-model and its accuracy, and to estimate BER before deciding on the mode of operation.

6 Research issues

1. *Distributed voting mechanism*
   Typical clustering algorithms (e.g., GAF) use energy as the sole criteria in electing cluster-heads. In Buddy protocol, BER is also an important factor. The energy costs in the two modes is influenced by BER on the link to the cluster-head (section 5.4.2). In the worst-case if the chosen cluster-head is such that a number of nodes experience high BER, they would be forced to operate in DEFAULT mode, thereby wasting energy. Ideal cluster-head is a node to whom other nodes in the cluster have very good link quality. In network with symmetric links, the BER on links from neighbors to a node can be approximated by BER on links from a node to its neighbors. Such an information may be collected locally. However, in sensor networks high degree of asymmetry in link characteristics has been observed [11]. In such a case, electing cluster-head would require a distributed voting mechanism. Implementing such a mechanism with small energy overhead is an important research issue.

2. *Cluster-head election mechanism*
   A cluster-head election mechanism should also take into account the expected predictability of readings of sensor nodes. In electing cluster-head, a node that is *not* expected to have predictable readings should be given preference over a node that is expected to have predictable readings. When an "unpredictable" node is made the cluster-head, it gives opportunities to "predictable node" to switch to PREMON mode, thereby saving energy. On the other hand, if a predictable node is made the cluster-head, it will not be able to save energy. Summarizing, cluster-head election should take into account the following three factors: energy, predictability, and BER (as explained above). Devising such a mechanism that is simple and has small energy overhead is a challenging research problem.

3. *How to exploit spatial correlation?*
   In this paper, we discussed exploiting temporal correlation in readings of a sensor. A research challenge is to extend this technique for exploiting spatial correlation. One possible approach may be to exploit the "broader-view" of the cluster available at the cluster-head. Specifically, if a cluster-head observes high spatial correlation in the readings of the cluster-nodes, it may compress their readings before sending them to the monitoring entity. However, a more general approach would be to follow the PREMON paradigm by computing a spatial prediction-model and sending it to the monitoring entity.

4. *Tree folding*
   If one were to view the node participating in a monitoring operation as forming a tree, use of PREMON inside the cluster may be seen as folding the branches connecting the leaf node. Can this branch folding operation be extended further? Specifically, instead of placing the prediction-model at the cluster-head, can it be placed closer to the monitoring entity. The closer it is to the monitoring entity, the smaller the energy consumed in reporting readings to the monitoring entity. However, this increases the cost of sending the model and the cost of sending violations. The optimal point of placement of the model is a function of the violation probability. An interesting research issue is to devise a mechanism that changes the placement of the prediction model based on the violation probability in order to increase energy savings.

5. *Clustering mechanism that makes use of specific characteristics of Buddy protocol*
   In this paper, we have made use of existing techniques for clustering. A research challenge is to devise techniques that considers the predictability of nodes when clustering them together. Such techniques would arguably perform better than a technique that is blind to these parameters that influence the performance of Buddy protocol.

7 Conclusions

In this paper, we presented a novel idea for exploiting temporal correlation. We showed that neighboring sensor nodes can help reduce each other's energy consumption by entering into collaborative buddy relationships. We presented a simple Buddy protocol that implements this idea. We analyzed its performance and identified factors that influence it.

This protocol shows one possible way in which the above idea can be implemented. We discussed research challenges that would pave the way for more efficient protocols based on the idea.

References

[1] IEEE Std. 802.11a - Wireless LAN Medium Access Control (MAC) and Physical layer (PHY) layer specifications: High Speed Physical Layer in the 5 GHz Band, 1999.

[2] G. Anastasi, M. Conti, A. Falchi, E. Gregori, and A. Passarella. Performance measurements of mote sensor networks. In *Proceedings of the ACM/IEEE Symposium on Modeling, Analysis and Simulation of Wireless and Mobile System (MSWIM 2004)*, October 2004.

[3] Samir Goel and Tomasz Imieliński. Prediction-based monitoring in sensor networks: Taking lessons from mpeg. *ACM Computer Communication Review*, 31(5), October 2001.

[4] C. Guestrin, P. Bodik, R. Thibaux, M. Paskin, and S. Madden. Distributed regression: an efficient framework for modeling sensor network data. In *Proceedings of the Third International Symposium on Information Processing in Sensor Networks (IPSN '04)*, April 2004.

[5] J. Hill, R. Szewczyk, A. Woo, S. Hollar, and K. Pister D. Culler. System architecture directions for networked sensors. In *Proceedings of the 9th Int'l Conf. on ASPLOS*, November 2000.

[6] Mani Srivastava. Energy efficient wireless systems. In *DIMACS Summer School on Foundations of Wireless Networks and Applications*, August 2000.

[7] R. Szewczyk, J. Polastre, A. Mainwaring, and D. Culler. Lessons from a sensor network expedition. In *Proceedings of the First European Workshop on Wireless Sensor Networks (EWSN '04)*, January 2004.

[8] Y. Xu, S. Bien, Y. Mori, J. Heidemann, and D. Estrin. Topology control protocols to conserve energy in wireless ad hoc networks. Technical Report 0006, CENS, January 2003.

[9] Ya Xu, John Heidemann, and Deborah Estrin. Geography-informed energy conservation for ad-hoc routing. In *Proceedings of 7th Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom '01), Los Angeles*, July 2001.

[10] W. Ye, J. Heidemann, and D. Estrin. An energy-efficient mac protocol for wireless sensor networks. In *Proceedings of IEEE INFOCOM'02*, June 2002.

[11] Jerry Zhao and Ramesh Govindan. Understanding packet delivery performance in dense wireless sensor networks. In *Proceedings of the first International Conference on Embedded networked sensor systems*. ACM Press, 2003.